ISSN 2409-2665
Journal of Logistics, Informatics and Service Science
Vol. 10 (2023) No. 4, pp.154-173
DOI:10.33168/JLISS.2023.0411

Enhancing MySQL Database Security with MySQL Enterprise Transparent Data Encryption

Nathanael Terencio, Wella, Antonius Sony

Faculty of Engineering and Informatics, Universitas Multimedia Nusantara, Indonesia

Abstract. With the increasing threats to data security and the potential consequences of data breaches, the demand for data security has steadily risen. Protecting sensitive information, particularly stored in databases, has become a crucial aspect of data management. Considering this, this study focuses on a specific solution for data protection within databases, MySQL Transparent Data Encryption, specifically in the case of PT ABC, an information and technology company. MySQL Transparent Data Encryption was implemented as a potential solution to enhance data security. The implementation of MySQL Transparent Data Encryption was tested by benchmarking to evaluate the effectiveness and efficiency of the solution before and after the implementation using JMeter, a widely recognized and reliable tool for performance testing. The research findings demonstrate that implementing MySQL Transparent Data Encryption could effectively secure the database while having a minimal impact on performance. This is a significant finding, as it shows that enhanced data security only sometimes comes at the cost of reduced performance. The encryption implementation resulted in a less than 10% decrease in database performance, indicating that the advantages of securing data outweigh the minor performance decrease. In conclusion, this study confirms the reliability of MySQL Transparent Data Encryption as a solution for securing data within a MySQL database. It confirms that this method of encryption is effective in enhancing data security and efficient in maintaining the performance of the database.

Keywords: Database, Security, Protection, MySQL, Transparent Data Encryption, Benchmarking

1. Introduction

In today's digital era, organizations face increasing challenges in protecting their sensitive data from unauthorized access and potential breaches. With the increase in unauthorized access and potential breaches, organizations must adopt a reliable security tool to safeguard their data in the database. Reliable security tool adoption aims to protect an organization's internal data in real time (Al-Zahrani & Al-Hebbi, 2022). One of the reliable tools that can be adopted is Transparent Data Encryption (TDE) in database management systems. This article presents a comprehensive case study on implementing MySQL Transparent Data Encryption to enhance database security, focusing on the use case of PT ABC, one of the technology companies providing IT solutions in Indonesia. With a vast customer base and a wide range of IT products and services, PT ABC handles sensitive data. As a responsible organization committed to data security, PT ABC recognizes the importance of implementing strong security measures to protect sensitive data. Sensitive data is usually stored as data at rest (Vicarte et al., 2022). PT ABC uses MySQL as its database management system. Transparent Data Encryption in MySQL is a feature that provides an additional layer of security by encrypting data at rest, ensuring that even if an attacker gains unauthorized access to the underlying storage media, the data remains encrypted and unreadable by the attacker (MySQL, 2023). Transparent Data Encryption encrypts data files, log files, and temporary files associated with the MySQL database, which ensures the protection of sensitive data from unauthorized access.

PT ABC will achieve several key benefits by implementing MySQL Transparent Data Encryption for their database. Firstly, it ensures compliance with industry regulations and data protection standards, such as ISO (International Organization for Standardization). Compliance with these regulations is crucial for PT ABC to maintain its reputation. Secondly, implementing MySQL Transparent Data Encryption provides additional security for PT ABC's data, mitigating the risk of unauthorized access and data breaches. With an additional encryption layer in place, even if an attacker gains access to the physical storage media, the encrypted data remains unreadable without the appropriate decryption keys, managed by MySQL TDE's key management. The added security layer will significantly reduce the potential impact of a data breach and help protect PT ABC's stored sensitive information. Thirdly, Transparent Data Encryption offers peace of mind to PT ABC by ensuring that even if MySQL experiences a security incident, the encrypted data at rest remains protected.

While the benefits of implementing MySQL Transparent Data Encryption are significant, the challenges and considerations to address during the implementation process must be acknowledged. One of the challenges and considerations is the potential impact on system performance. The encryption and decryption processes can introduce additional overhead to the system, affecting overall performance. Optimize the system accordingly to ensure a seamless user experience. To ensure that the implementation can overcome this challenge, this research will conduct benchmarking to test the impact of implementing MySQL Transparent Data Encryption. Therefore, this research aims to investigate the results of the protection method using MySQL Transparent Data Encryption (TDE) due to the limited research utilizing this method. In this research, MySQL TDE tools will be implemented on PT ABC's database to provide protection and enhance the security of the database, thus mitigating existing risks. The case study in this research will delve into the technical details of the implementation process, explore the benefits and challenges PT ABC faces, and provide practical recommendations for organizations to consider implementing MySQL Transparent Data Encryption to enhance their data security. There are also some research questions to be solved:

RQ1: How can the results of database encryption help PT ABC protect company data from data security risks?

RQ2: How does the encryption performance result on the database affect the overall system performance?

There are also some limitations to this research:

LM1: Database protection is carried out using MySQL TDE.

LM2: The DBMS system used is MySQL.

LM3: The protection that will be implemented will be tailored according to the needs of PT ABC.

2. Literature Review

2.1. Database

A database is a collection of data or information stored in a computer system that systematically forms a file (Yani & Saputra, 2018). A database can also be defined as a collection of computerized files or as a computerized system that serves as an information storage facility that can provide data when needed (Yani & Saputra, 2018; Gosal & Rustam, 2022). A database can also be defined as a collection of useful data and information that is organized in a special arrangement and can be used according to the needs of an organization (Yani & Saputra, 2018; Seputar Pengetahuan, 2017). Databases represent a set of data and information stored in a unit without unnecessary reduction, which can meet various needs. In this context, a database can be defined as a collection of data and information interrelated on a specific subject with a specific purpose, which forms a set of records of data of an organization or company, organized in a special arrangement that is organized and integrated using a specific method in a computer system that can optimally meet the information needs when needed by an organization or company (Helmud, 2021). Databases become the primary need of an organization or company to be able to store and organize the important data they have.

2.2. Database Protection

Data protection is a form of individual data protection concerning personal data stored in a computer system (Collins Dictionary, 2023). Data protection is a form of protection that includes data collection, data distribution, and technology. Data protection is known as information privacy. Data protection should apply to all forms of data, whether individual or corporate. Data protection also has to do with data integrity, where data can only be accessed by people with access rights. Data protection generally means protecting data from unauthorized access to prevent it from falling into the hands of others other than its owners (Elmasri et al., 2017). Databases as data storage sites should also be protected to avoid the risk of data leakage or theft from unauthorized access. Database protection is closely related to intentional and unintentional threats. Risks and threats to databases can reduce or eliminate the objectives of database security, such as data integrity, availability, and confidentiality (Elmasri et al., 2017). Database protection is not only about the data contained in the database itself but also about the security of other parts that affect the database's security, such as networks, operating systems, buildings where the database is physically located, and people who have access to the system (Susilo, 2016). In organizations such as companies, data security is taken care of; the confidentiality of data held by the company must be protected from the risks and threats to data security. Risks such as data leaks and theft can be avoided by protecting the data and databases held, one of which can use encryption methods (Wiharto & Irawan, 2018).

2.3. Encryption

Encryption is a technical process that transforms data or information into code, making it unreadable in its original form. Encryption can also be understood as scrambling data before decryption is performed to read the encrypted data or information. Data or information that is not encrypted is referred to as "plaintext," while data or information that has been encrypted is known as "cipher text" (Suhaemin & Muslih, 2023). From this definition, encryption can also be seen as converting plaintext into ciphertext. Encryption is a crucial component of cryptography and plays a vital role in maintaining the confidentiality of data or information (Sibyan, 2017). Encrypting a database can be a solution to protecting the data within it. For organizations, especially companies, encrypting databases is an appropriate solution to safeguard important data from leaks and theft. Database encryption can be defined as a process that utilizes a specific algorithm to modify the data within a database into a secret code that cannot be read without decryption (Manohar et al., 2020). Encrypting a database reduces the threat of database breaches since the data within the database is already encrypted (Manohar et al., 2020). (Steele et al., 2020). Encrypting a database ensures that anyone with malicious intent to breach or steal the data cannot read the data therein. This unreadable data holds no value, diminishing individuals' motivation to breach or steal data from the database (Shiyal, 2021).

2.4. Benchmarking

Benchmarking is a commonly used method for measuring the performance and capabilities of a system. In the context of databases, benchmarking is conducted to assess the performance and capabilities of a database in managing and storing data (Billah, 2020). The purpose of database benchmarking is to gather information about the speed and efficiency of a database system in handling given workloads (Stragapede et al., 2023). Benchmarking is typically performed by creating a series of tests or scenarios applied to the system under evaluation. The benchmarking process produces various measurable indices, including response time, latency, system resource utilization, and others (MySQL, 2023) (Mudrikah, 2022). Benchmarking can be categorized into different types, including functional benchmarking and performance benchmarking. Functional benchmarking focuses on testing the system's functionality under evaluation (Ali et al., 2022) (MySQL, 2023). On the other hand, performance benchmarking evaluates the database's performance in handling given workloads with speed and efficiency (Ali et al., 2022) (MySQL, 2023). Benchmarking, particularly in the context of databases, plays a crucial role in developing and maintaining database systems, as it assists organizations or companies in making informed decisions.

2.5. MySQL

MySQL is a database management system (DBMS) run in structured query language (SQL) widely used by system developers in developing systems. MySQL is ranked 2nd in the Database Management Systems DB-Engines Ranking based on popularity, only behind Oracle (DB-Engines, 2023). MySQL is also included as the Relational Database Management System (RDBMS), which processes data collection using the method of relational databases (Kurniawan, 2020). MySQL has many features, such as security, data integrity, backup and restore, and even database replication. MySQL is commonly used as a database to build web applications, desktops, and other systems in system development. MySQL can also be integrated with various programming languages such as Python, PHP, Java, Etc. (Adi, Puput, & Akio, 2018).

2.6. MySQL Transparent Data Encryption

MySQL Transparent Data Encryption (TDE) is one of the security features provided in MySQL Enterprise Edition (EE), enabling users to encrypt data within the database transparently. MySQL TDE utilizes the Advanced Encryption Standard (AES), a well-established and secure encryption algorithm (MySQL, 2023). By encrypting data using the AES algorithm, MySQL TDE allows users of MySQL EE to enhance the security of their data stored in the database. Furthermore, MySQL TDE enables users to individually encrypt columns or tables within the database, granting them the flexibility to select specific data for encryption. Additionally, MySQL TDE supports integration with MySQL Enterprise Backup, ensuring secure backup data encryption (MySQL, 2023). Leveraging database caching, TDE achieves high-performance capabilities (MySQL, 2023). By implementing MySQL TDE, users can ensure the safety of their data from threats such as leaks or data theft (Ali et al., 2022). MySQL TDE offers several advantages concerning data security and protection. The data protection provided by MySQL TDE safeguards the data stored in the database against unauthorized access from internal and external sources. Through robust encryption, stolen or illegally accessed data becomes unreadable and unusable by unauthorized entities. Moreover, MySQL TDE allows users to encrypt desired tables and columns selectively, effectively preserving system performance (MySQL, 2023).

2.7. JMeter

JMeter is software that can be used to test or benchmark web-based systems. The software was developed by the Apache Software Foundation, which provides an environment that can be used to perform benchmarking on systems, including databases (Mudrikah, 2022).

3. Research Method

3.1. Research Framework

The research will be conducted in five stages: problem identification, protection technique selection, Protection Technique implementation, implementation result testing, and Evaluation.

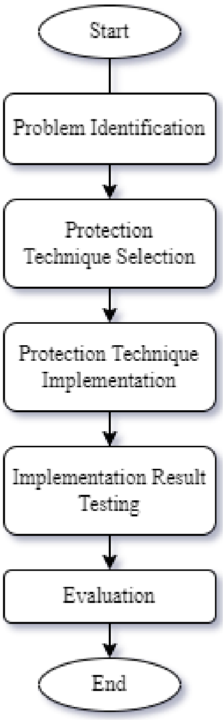

Start
Problem Identification
Protection Technique Selection
Protection Technique Implementation
Implementation Result Testing
Evaluation
End

Fig.1: Conceptual Framework

a. Problem Identification:
The research begins with identifying problems faced by the database at PT ABC. Problem identification will be conducted by interviewing representatives from PT ABC, specifically the head of solutions architecture and competency development. The interview results will reveal the problems faced by the company related to the database used. Once the problems have been successfully identified, the research will proceed to the next stage, which is the technique selection phase, to explore various methods that can be used to address the problems identified in the previous stage. In this stage, we will identify the problems faced by PT ABC related to its MySQL database and determine potential solutions to address these issues. Some problems that PT ABC may experience regarding its database include data leakage or theft resulting from system vulnerabilities that can be exploited by irresponsible parties, both internally and externally.

b. Protection Technique Selection:
The next step is to select protection techniques after identifying the problems in the problem identification phase. Various database protection methods will be compared in this phase to address the identified problems. Some protection techniques that can be applied to address the potential issues faced by PT ABC related to their database include data encryption, strong passwords, access restrictions to the database, or employing database encryption tools provided by MySQL, such as MySQL TDE. The selection of protection techniques should be based on a thorough analysis of the data security in PT ABC's database. The researcher must consider the required level and form of security PT ABC needs and the system's capabilities to

implement the chosen protection technique. After comparing, one protection method will be chosen for the research and implemented. The selected protection method from this phase will be implemented in the next phase.

c. Protection Technique Implementation:
After selecting the appropriate protection techniques in the second phase, the chosen protection techniques will be implemented in the technique implementation phase. The implementation will be carried out for PT ABC's database. The implementation will take place in a live server environment. The company's database and web will be deployed to the prepared live server environment. The protection techniques must be implemented carefully and attentively to ensure smooth operation and avoid potential new issues. In this phase, the researcher must pay close attention to the configuration and settings of the selected protection techniques to ensure effective and efficient operation. The researcher must also ensure that the implemented protection techniques do not disrupt the existing system's performance. Once the selected protection methods have been successfully implemented, the implementation results will be tested. During the implementation phase, the researcher must ensure that the implemented protection techniques function properly and effectively protect the data stored in PT ABC's database.

d. Implementation Result Testing:
After successfully implementing the protection techniques in the technique implementation phase, the next step is to test the implemented techniques in the implementation results testing phase. In this phase, the implementation results will be tested for their security. The researcher will test the PT ABC database with the previously implemented protection techniques. The testing phase will also involve benchmarking to assess how the chosen protection methods affect the performance of the company's database system. The testing conducted in this phase will cover several aspects, including data security testing, data access speed testing, and testing against external attacks. The testing will involve simulations of encryption. The results of these tests will provide an overview of the success and effectiveness of the protection techniques implemented in the previous phase. The test results will then be evaluated. If the implementation results are ineffective, adjustments or improvements may be required for the implemented protection techniques. The testing process will use JMeter as a benchmarking tool and test the database system's load time, connect time, and latency to present the database performance before and after the implementation of MySQL Transparent Data Encryption. This testing process will explain how MySQL Transparent Data Encryption impacts database performance.

e. Evaluation:
After the completion of the testing phase, an evaluation will be conducted to consider and compare the strengths and weaknesses of the implemented protection system, allowing future research to address the identified shortcomings. The evaluation phase serves as the final stage of this research. In this phase, the researcher will evaluate the results of the research and implementation that have been conducted. The purpose of the evaluation is to assess the success of the implemented protection techniques and how they affect the system's performance. During this evaluation phase, the researcher will analyze the results from the testing phase and compare them to the predetermined research objectives. If the implementation results successfully protect the database, it can be concluded that the research objectives have been achieved. However, if the testing results do not demonstrate satisfactory outcomes, further analysis will be required to determine the appropriate course of action. The evaluation results will provide

recommendations for future research.

3.2. Data Collection

The method of data collection that will be used in this study is the interview method. The interview method will be carried out to gather data and information about any problems the company faces related to the database and what can be done to solve those problems. In this study, the interviewed party is the Head of Solutions Architect and Competency Development at PT ABC, a database expert with various certifications. The interview is done online through Zoom Meeting, Google Meet, or directly by appointment with interested parties.

4. Result and Discussion

The research carried out the 5 phases as follows:

4.1. Problem Identification

Problem identification is the initial step undertaken. During the problem identification phase, interviews were conducted with the Head of Solutions Architect and Competency Development from PT ABC to ascertain the issues faced by PT ABC regarding its database. The interviews revealed that PT ABC has recently obtained ISO 27001 and ISO 9001 certifications, indicating the company's commitment to ensuring high-quality security and service. As a company operating in the Information and communication technology industry, PT ABC stores and manages several sensitive and critical pieces of data in its corporate database. During the interview, the Head of Solutions Architect and Competency Development from PT ABC also explained that the rapid development of the Information and communication technology industry has made data security a crucial and essential aspect for all companies. PT ABC acknowledges that in its efforts to enhance existing data security, database protection plays a vital role in safeguarding the confidentiality and integrity of sensitive data and information stored within the company's database. With robust database protection, the company's data security will have stronger defense mechanisms against threats like data breaches or theft. During the interview, the Head of Solutions Architect and Competency Development from PT ABC also highlighted specific challenges faced by PT ABC regarding database availability and overall system performance. The company must prioritize the speed of access and efficiency of the existing system since business operations rely on effective and accurate data management and processing. Therefore, careful consideration must be given to the protection techniques for PT ABC's MySQL database. The protection techniques should provide high security while maintaining optimal and efficient system performance.

By conducting this problem identification, a clear understanding of PT ABC's requirements regarding their database security and the challenges faced by the company concerning database and data security is obtained. The problem identification serves as the foundation for the database protection to be implemented for PT ABC's MySQL database, where the applied protection techniques must fulfill the company's security needs, including data-at-rest protection (data not being accessed or used), as well as considering system performance efficiency and database availability that will be protected.

4.2. Protection Technique Selection

A literature review is conducted after identifying the problem in the problem identification phase to select a protection technique. The literature review aims to identify the factors that contribute to the potential occurrence of data leaks. The literature review found that the AES algorithm effectively and efficiently provides data protection within the database. It was also discovered that MySQL, the database management system used by the company, is suitable for managing Big Data and has good usability. The literature review further revealed that effective key management is a critical factor in data protection through encryption. Proper management of technology, as one of the factors contributing to data breaches, is crucial for data protection. Therefore, this study will utilize a data protection tool that applies the AES algorithm for data encryption and good key management to ensure data protection. Transparent Data Encryption is one such tool that employs the AES algorithm for encryption. Transparent Data Encryption in MySQL protects by encrypting data at rest, encrypting the physical files of the database, and using a centralized key management system where all encryption keys are stored and managed in one location. In this research, data-at-rest stored in the company's MySQL database, such as customer data or financial data typically stored for the long term and not in transit over a network, will be protected. Given the need for data-at-rest protection and the utilization of the AES algorithm in the chosen protection technique, MySQL Transparent Data Encryption is the primary option for this study. Previous research has proven that transparent data encryption is the best feature for data protection in databases. However, prior studies have also indicated that implementing TDE features, such as CPU usage, memory usage, or backup duration, may impact system performance.

Nevertheless, the benefits derived from this feature outweigh these considerations when providing data protection. Over time, TDE features have been enhanced to be more performance-friendly, as seen in the significant improvements made by Oracle in their latest version, particularly in terms of CPU and storage.

Transparent Data Encryption, commonly known as TDE, is an encryption feature that allows users to encrypt sensitive data stored in a table or table space. TDE transparently encrypts data and performs real-time decryption for authorized users or applications. TDE serves as a data protection feature for data-at-rest, referring to data that is not actively used or transferred and is typically stored for long periods with minimal changes. Encryption techniques come with challenges, particularly managing encryption keys. A two-tier encryption key architecture is employed by utilizing Transparent Data Encryption, especially in MySQL databases, consisting of a master encryption key and table space keys. This architecture provides easy key management and rotation, with table space keys automatically managed through protocols while the master encryption key is stored centrally in the key management system. With TDE, even if unauthorized individuals gain access to the master key, the encrypted data remains inaccessible without the necessary access to the key management. In a physical data breach or theft, the data cannot be decrypted without the required keys. A comparison of several techniques studied in the literature review is conducted to ensure the selection of an appropriate protection technique, considering aspects such as security, performance efficiency, compatibility, and flexibility. The following table presents a comparison of various protection techniques identified in the literature review:

Table 1: Protection Technique Comparison

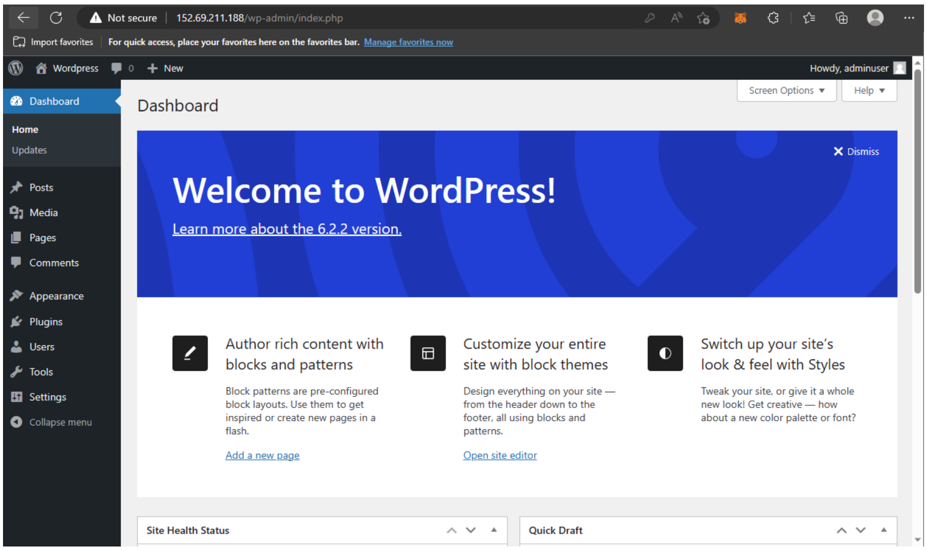
Protection Technique	Security	Efficiency	Flexibility
AES	AES is a widely used and recognized encryption standard for its security.	AES is extremely efficient and fast in data encryption and decryption.	AES supports three key sizes: 128, 192, and 256 bits that enable a level of security to suit the needs.
End-to-end symmetric	Symmetrical end-to-end encryption protects data during transmission between sender and recipient.	The efficiency of end-to-end symmetrical encryption depends on the algorithm used (AES, DES, or 3DES).	The flexibility of end-to-end symmetrical encryption depends on the algorithm used and the mode of operation chosen.
Transparent Data Encryption	MySQL TDE uses the AES (Advanced Encryption Standard) algorithm for data	Implementing these encryption and decryption processes can add a burden on the	TDE is quite flexible in terms of configuration. For example, users can choose which tables to

Protection Technique	Security	Efficiency	Flexibility
	encryption, which has proven good in protecting data.	system and potentially affect performance.	encrypt, the key size used, and whether the key rotation should be enabled.

Table 1 compares several protection techniques identified in the literature review: the AES algorithm, end-to-end symmetric encryption, and Transparent Data Encryption. From the table, it can be observed that Transparent Data Encryption (TDE) offers advantages. TDE utilizes the well-established AES algorithm for data protection. However, implementing this tool may introduce additional system overhead and impact overall system performance.

4.3. Protection Technique Implementation

During the implementation of the protection technique, the first step is to set up the MySQL Enterprise Edition database environment, which will serve as the implementation environment for the selected protection technique. The environment setup begins by installing MySQL Enterprise Server, MySQL Enterprise Shell, and MySQL Enterprise Router. These three MySQL Enterprise infrastructures are installed to ensure that MySQL Enterprise Edition can be used within the created environment. Installing MySQL Enterprise Server, MySQL Enterprise Shell, and MySQL Enterprise Router ensures that MySQL Enterprise Edition can be utilized in the environment intended for implementing the protection technique. Once these three components are installed, the MySQL database is set up to be used. Subsequently, Apache and PHP are installed to set up WordPress as the virtual environment to conduct the implementation. With the successful installation of Apache and PHP, the installation and configuration of WordPress can be performed to set up the environment that will be used for implementing MySQL Transparent Data Encryption. WordPress is created with a dedicated database that has been prepared called 'web_ict'. WordPress will serve as the live environment used to test the implementation of the selected protection technique. The following image depicts the setup of WordPress that has been carried out:



For quick access, place your favorites here on the favorites bar. Manage favorites now

Dashboard

Welcome to WordPress!

Learn more about the 6.2.2 version.

Author rich content with blocks and patterns

Block patterns are pre-configured block layouts. Use them to get inspired or create new pages in a flash.

Customize your entire site with block themes

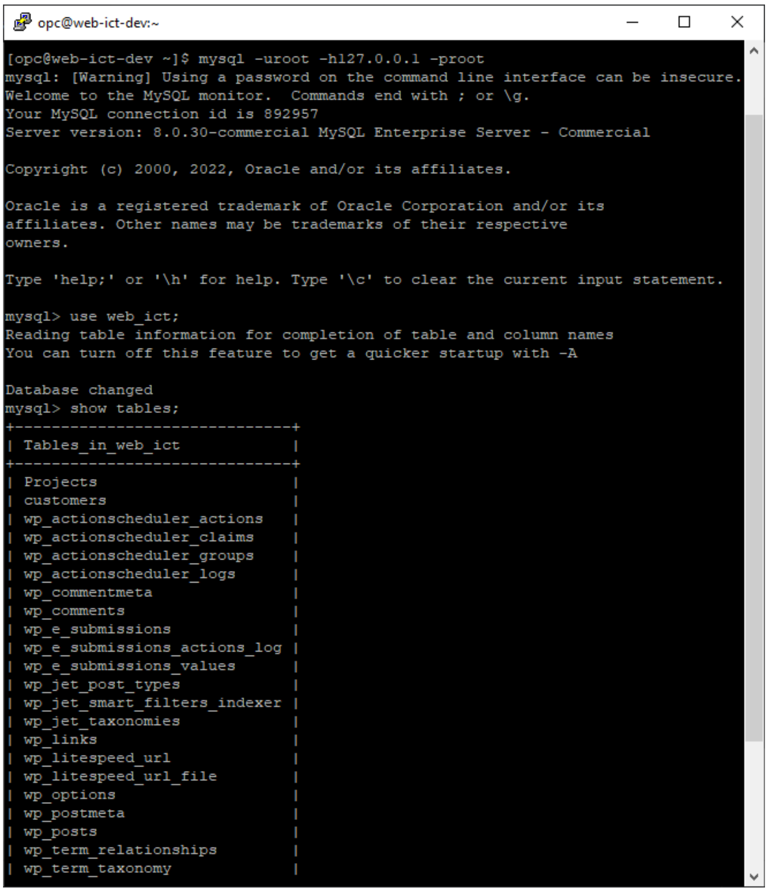
Design everything on your site — from the header down to the footer, all using blocks and patterns.

Switch up your site's look & feel with Styles

Tweak your site, or give it a whole new look! Get creative — how about a new color palette or font?

Fig.2: WordPress Set Up Result

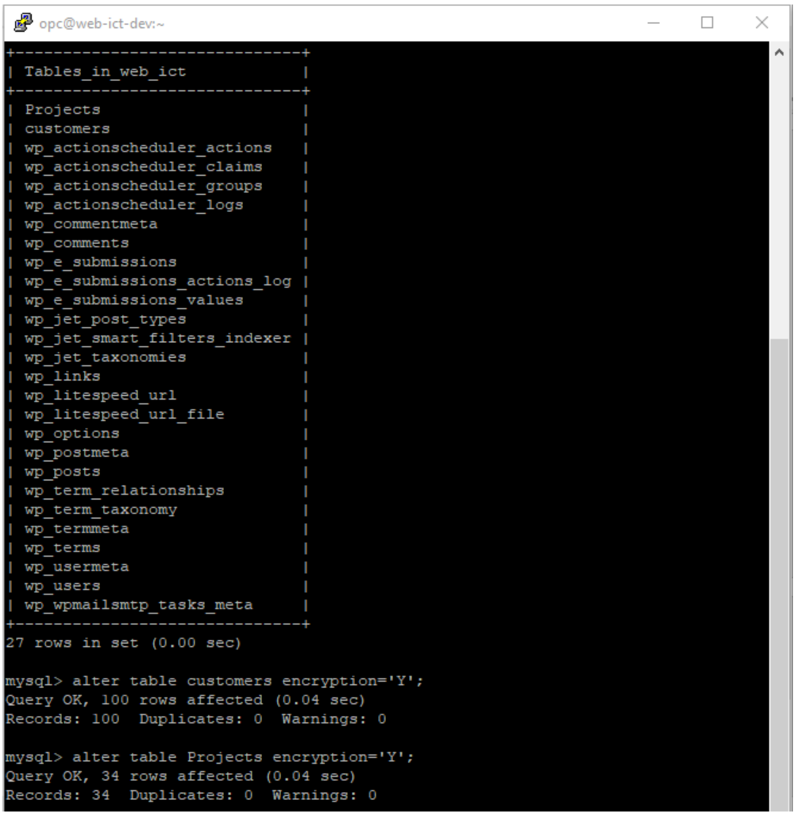
Figure 2 shows the result of the installation and setup of WordPress. WordPress has been configured to use the 'web_ict' database created earlier. WordPress will be used as the environment for implementing MySQL Transparent Data Encryption. After the environment's successful setup, the protection techniques are implemented, starting with the implementation of MySQL TDE. The following image illustrates the encryption implementation using MySQL TDE for table-level encryption.



```
[opc@web-ict-dev ~]$ mysql -uroot -h127.0.0.1 -proot
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 892957
Server version: 8.0.30-commercial MySQL Enterprise Server - Commercial

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use web_ict;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+------------------------------+
| Tables_in_web_ict            |
+------------------------------+
| Projects                     |
| customers                    |
| wp_actionscheduler_actions   |
| wp_actionscheduler_claims    |
| wp_actionscheduler_groups    |
| wp_actionscheduler_logs      |
| wp_commentmeta               |
| wp_comments                  |
| wp_e_submissions             |
| wp_e_submissions_actions_log |
| wp_e_submissions_values      |
| wp_jet_post_types            |
| wp_jet_smart_filters_indexer |
| wp_jet_taxonomies            |
| wp_links                     |
| wp_litespeed_url             |
| wp_litespeed_url_file        |
| wp_options                   |
| wp_postmeta                  |
| wp_posts                     |
| wp_term_relationships        |
| wp_term_taxonomy             |
```

Fig.3: Tables in ‘web_ict’ database

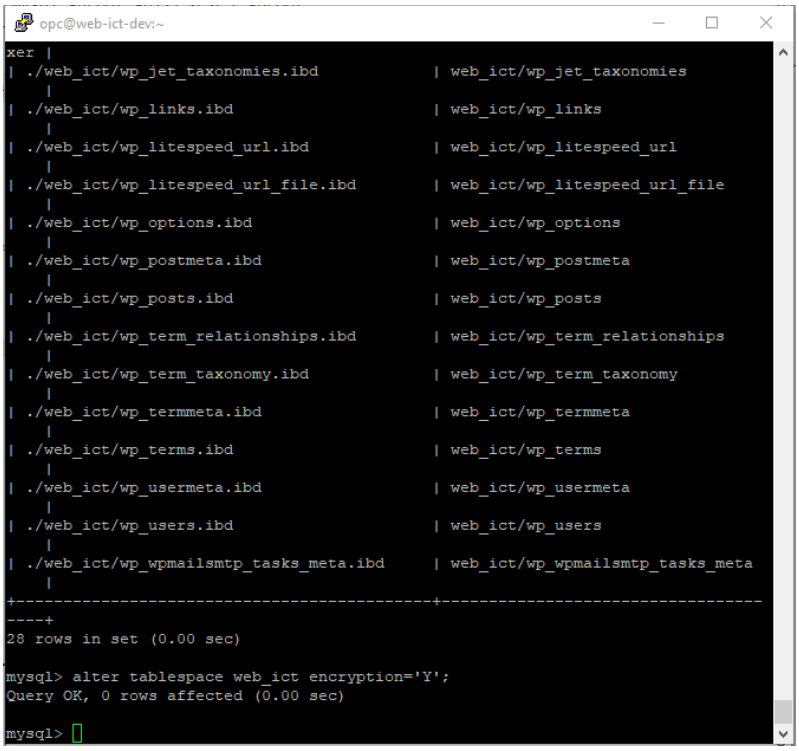
Figure 3 shows the verification process carried out for the tables in the database ‘web_ict, which contains the data on the website of PT ABC. It shows the tables Projects, Customers, wp_actionscheduler_acts, wp_commentmeta, wps_comments, Etc. Projects and customers tables contain dummy data from customer data and project data run by PT ABC. For table-level testing, encryption will be performed for the Projects and Customers tables that represent the data-at-rest owned by the company. Here is the encryption process for the Projects and customer tables:



```
+------------------------------+
| Tables_in_web_ict            |
+------------------------------+
| Projects                     |
| customers                    |
| wp_actionscheduler_actions   |
| wp_actionscheduler_claims    |
| wp_actionscheduler_groups    |
| wp_actionscheduler_logs      |
| wp_commentmeta               |
| wp_comments                  |
| wp_e_submissions             |
| wp_e_submissions_actions_log |
| wp_e_submissions_values      |
| wp_jet_post_types            |
| wp_jet_smart_filters_indexer |
| wp_jet_taxonomies            |
| wp_links                     |
| wp_litespeed_url             |
| wp_litespeed_url_file        |
| wp_options                   |
| wp_postmeta                  |
| wp_posts                     |
| wp_term_relationships        |
| wp_term_taxonomy             |
| wp_termmeta                  |
| wp_terms                     |
| wp_usermeta                  |
| wp_users                     |
| wp_wpmailsmtp_tasks_meta     |
+------------------------------+
27 rows in set (0.00 sec)

mysql> alter table customers encryption='Y';
Query OK, 100 rows affected (0.04 sec)
Records: 100  Duplicates: 0  Warnings: 0

mysql> alter table Projects encryption='Y';
Query OK, 34 rows affected (0.04 sec)
Records: 34  Duplicates: 0  Warnings: 0
```

Fig.4: Encryption for tables ‘customers’ and ‘Projects’

Figure 4 shows the encryption process performed for the Projects and customers tables. The Projects table contains dummy data from projects carried out by PT ABC, whereas the Customers table contains dummy information from company customers. Both tables represent data-at-rest owned by the company. Table-level encryption using MySQL TDE is done with the command "alter table <table-name> encryption=’Y';". This command will encrypt the chosen table. A test for encryption in table space levels was also carried out to better test the performance test. Here is a test of encryption at the table space level for the "web_ict" tablespaces:



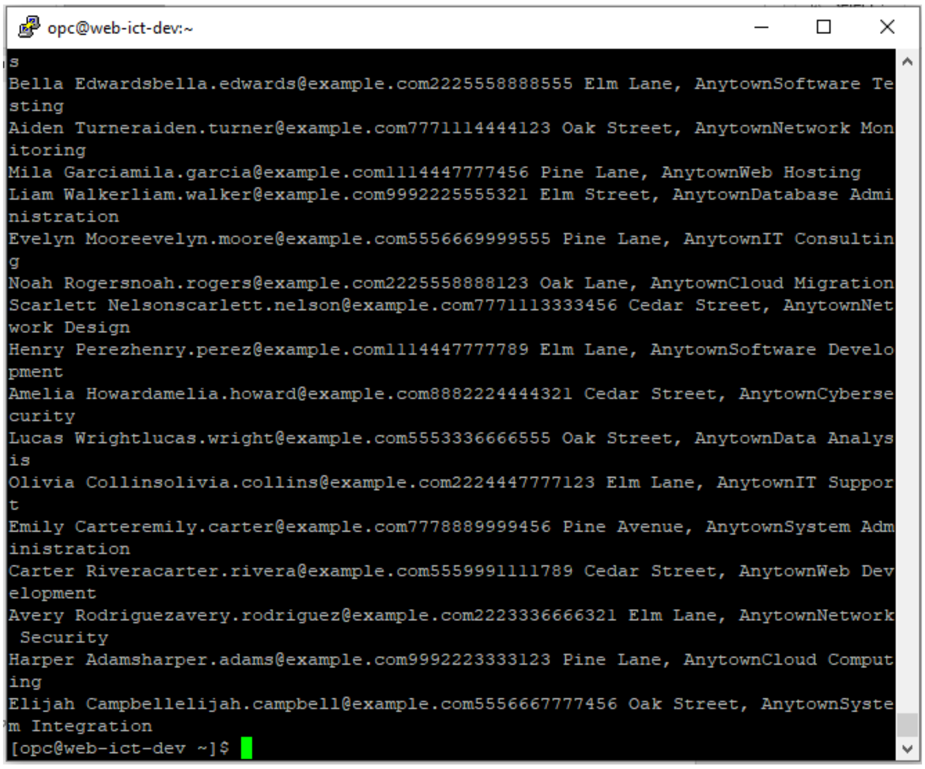
```
xer |
| ./web_ict/wp_jet_taxonomies.ibd           | web_ict/wp_jet_taxonomies
    |
| ./web_ict/wp_links.ibd                    | web_ict/wp_links
    |
| ./web_ict/wp_litespeed_url.ibd            | web_ict/wp_litespeed_url
    |
| ./web_ict/wp_litespeed_url_file.ibd       | web_ict/wp_litespeed_url_file
    |
| ./web_ict/wp_options.ibd                  | web_ict/wp_options
    |
| ./web_ict/wp_postmeta.ibd                 | web_ict/wp_postmeta
    |
| ./web_ict/wp_posts.ibd                    | web_ict/wp_posts
    |
| ./web_ict/wp_term_relationships.ibd       | web_ict/wp_term_relationships
    |
| ./web_ict/wp_term_taxonomy.ibd            | web_ict/wp_term_taxonomy
    |
| ./web_ict/wp_termmeta.ibd                 | web_ict/wp_termmeta
    |
| ./web_ict/wp_terms.ibd                    | web_ict/wp_terms
    |
| ./web_ict/wp_usermeta.ibd                 | web_ict/wp_usermeta
    |
| ./web_ict/wp_users.ibd                    | web_ict/wp_users
    |
| ./web_ict/wp_wpmailsmtp_tasks_meta.ibd    | web_ict/wp_wpmailsmtp_tasks_meta
    |
+-------------------------------------------+----------------------------------
----+
28 rows in set (0.00 sec)

mysql> alter tablespace web_ict encryption='Y';
Query OK, 0 rows affected (0.00 sec)

mysql> 
```

Fig.5: Encryption for table space ‘web_ict’

Figure 5 shows the encryption process for table space "web_ict". In MySQL TDE, encryption at the table space level is performed with the command "alter table space tablespace-name> In MySQL TDE, encryption at the table space level is performed with the command "alter table space tablespace-name> encryption='Y';." This command will encrypt the table space separately, so all the data on the table will be encrypted. By performing encryption at the table space level, it will be possible to compare the impact of table-level encryptions and the table space levels that MySQL Transparent Data Encryption can do.

4.4. Implementation Result Testing

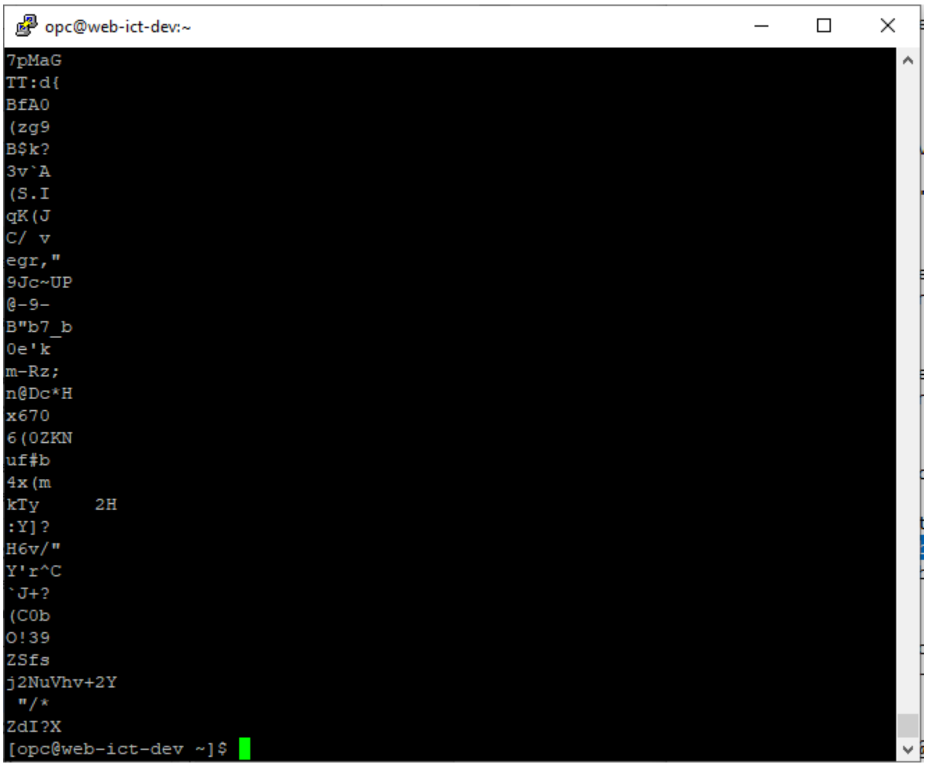
Database encryption is performed for tables in the database 'web_ict' using MySQL TDE. Implementing MySQL TDE protects data-at-rest, so to perform testing from the encryption that has been done, "Strings" against encrypted tables are used as a simulation when there is unauthorized access to the database to retrieve data. Here are the results of encryption with MySQL TDE implemented in the 'customers' table to represent table-level encryptions:



```
s
Bella Edwardsbella.edwards@example.com2225558888555 Elm Lane, AnytownSoftware Te
sting
Aiden Turneraiden.turner@example.com7771114444123 Oak Street, AnytownNetwork Mon
itoring
Mila Garciamila.garcia@example.com1114447777456 Pine Lane, AnytownWeb Hosting
Liam Walkerliam.walker@example.com9992225555321 Elm Street, AnytownDatabase Admi
nistration
Evelyn Mooreevelyn.moore@example.com5556669999555 Pine Lane, AnytownIT Consultin
g
Noah Rogersnoah.rogers@example.com2225558888123 Oak Lane, AnytownCloud Migration
Scarlett Nelsonscarlett.nelson@example.com7771113333456 Cedar Street, AnytownNet
work Design
Henry Perezhenry.perez@example.com1114447777789 Elm Lane, AnytownSoftware Develo
pment
Amelia Howardamelia.howard@example.com8882224444321 Cedar Street, AnytownCyberse
curity
Lucas Wrightlucas.wright@example.com5553336666555 Oak Street, AnytownData Analys
is
Olivia Collinsolivia.collins@example.com2224447777123 Elm Lane, AnytownIT Suppor
t
Emily Carteremily.carter@example.com7778889999456 Pine Avenue, AnytownSystem Adm
inistration
Carter Riveracarter.rivera@example.com5559991111789 Cedar Street, AnytownWeb Dev
elopment
Avery Rodriguezavery.rodriguez@example.com2223336666321 Elm Lane, AnytownNetwork
 Security
Harper Adamsharper.adams@example.com9992223333123 Pine Lane, AnytownCloud Comput
ing
Elijah Campbellelijah.campbell@example.com5556667777456 Oak Street, AnytownSyste
m Integration
[opc@web-ict-dev ~]$ 
```

Fig.6: Table ‘customers’ before encryption

Figure 6 shows the content of the data from the 'customers' table before being encrypted when performing 'Strings' on the table. Without encryption, the data could be stolen or leaked if an irresponsible party accesses the database. This data is then encrypted, as was done at the implementation stage of protection techniques. Here is a simulation of the "Strings" table for the Customers table after encryption:

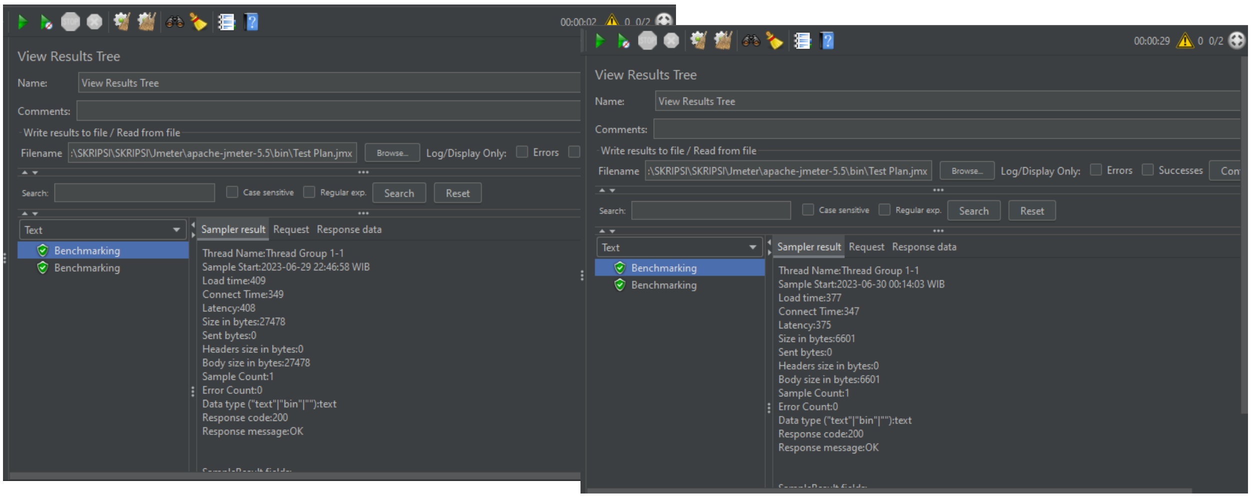


```
7pMaG
TT:d{
BfA0
(zg9
B$k?
3v`A
(S.I
qK(J
C/ v
egr,"
9Jc~UP
@-9-
B"b7_b
0e'k
m-Rz;
n@Dc*H
x670
6(0ZKN
uf#b
4x(m
kTy      2H
:Y]?
H6v/"
Y'r^C
`J+?
(C0b
O!39
ZSfs
j2NuVhv+2Y
 "/*
ZdI?X
[opc@web-ict-dev ~]$
```

Fig.7: Table 'customers' after encryption

Figure 7 shows the results of a data leak simulation using "Strings" on the encrypted "customers" table. Data from the 'customers' tables is well encrypted, so special access to the database is required to view the data from the tables. With the encryption done, if any irresponsible party accesses the database, the data will be protected from leakage and theft because it no longer has value. Without special access to the database as an authority, the data will be readable. MySQL TDE will perform transparent decryption for users who enter the database as the authority who has access.

To determine the impact of the implementation of protection techniques carried out on the performance of the system, benchmarking is performed using JMeter to look at the influence of implementation protection techniques on the 'load time', 'connect time, and 'latency' necessary to access and use the database. Benchmarking is divided into two levels: table space and table level. Benchmarking will be performed by performing queries "Select * From Customers" and "Selects * From Projects" as representatives of encryption at the table level. The query used for benchmarking shows a simulation of existing data-at-rest readings, where the data in the 'customers' and projects tables are to be read. Benchmarking is done to see the difference before and after encryption. Here are the results of the benchmarking performed for the database at the table level, for the 'customers' table, before encrypting and after encryption:


00:00:02 0 0/2
View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename :\SKRIPSI\SKRIPSI\Jmeter\apache-jmeter-5.5\bin\Test Plan.jmx
Browse...
Log/Display Only: □ Errors □
Search:
□ Case sensitive □ Regular exp.
Search
Reset
Text
Benchmarking
Benchmarking
Sampler result
Request
Response data

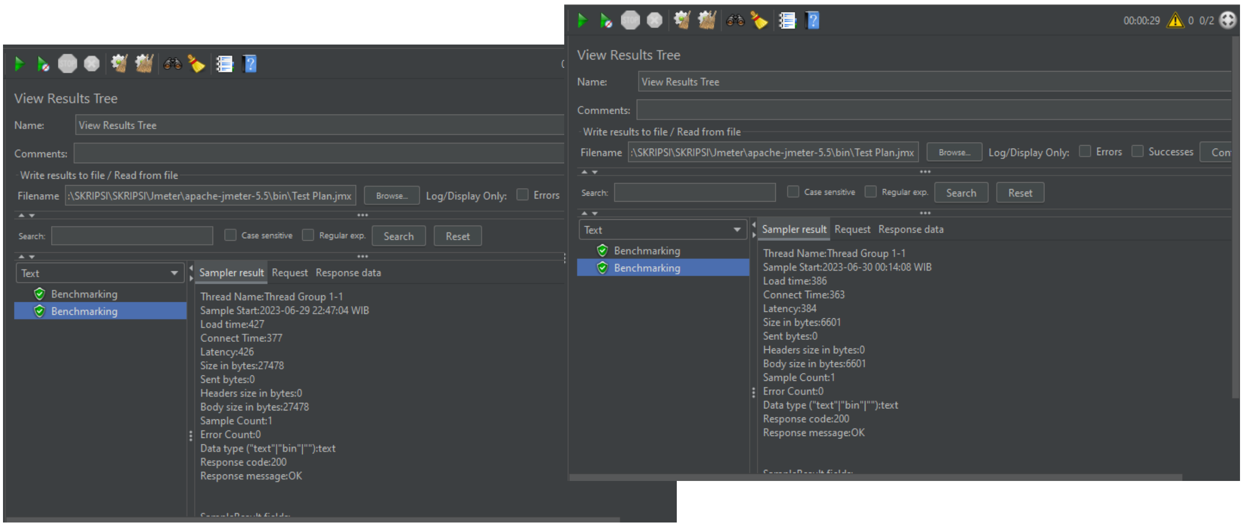
```
Thread Name:Thread Group 1-1
Sample Start:2023-06-29 22:46:58 WIB
Load time:409
Connect Time:349
Latency:408
Size in bytes:27478
Sent bytes:0
Headers size in bytes:0
Body size in bytes:27478
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:OK
```

00:00:29 0 0/2
View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename :\SKRIPSI\SKRIPSI\Jmeter\apache-jmeter-5.5\bin\Test Plan.jmx
Browse...
Log/Display Only: □ Errors □ Successes
Con
Search:
□ Case sensitive □ Regular exp.
Search
Reset
Text
Benchmarking
Benchmarking
Sampler result
Request
Response data

```
Thread Name:Thread Group 1-1
Sample Start:2023-06-30 00:14:03 WIB
Load time:377
Connect Time:347
Latency:375
Size in bytes:6601
Sent bytes:0
Headers size in bytes:0
Body size in bytes:6601
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:OK
```

Fig.8: Benchmarking for 'customers' and 'Projects' tables before encryption

Figure 8 shows the benchmarking results before implementing encryption protection techniques for the 'customers' tables. Load time is the time required to load the database, with data transactions of 27478 bytes, and is 409 ms. At the same time, connect time is the time necessary to connect to the database server and is 349 ms. Latency is the necessary time to transfer data between two points to measure network response speed and is 408 ms. The figure also shows the benchmarking results performed before the encryption protection techniques implementation for the 'Projects' table. Load time is the time required to load the database with data transactions of 6601 bytes and is 377 ms. At the same time, connect time is the time necessary to connect to the database server and is 347 ms. Latency is the necessary time to transfer data between two points to measure network response speed and is 375 ms. With smaller data sizes, load time, connect time, and latency have values like benchmarking for larger data sizes. This is because the computer resources used to perform benchmarking also affect load time, connect time and latency. Here are the benchmarking results for the 'customers' and projects tables after encryption:


View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename :\SKRIPSI\SKRIPSI\Jmeter\apache-jmeter-5.5\bin\Test Plan.jmx
Browse...
Log/Display Only: □ Errors
Search:
□ Case sensitive □ Regular exp.
Search
Reset
Text
Benchmarking
Benchmarking
Sampler result
Request
Response data

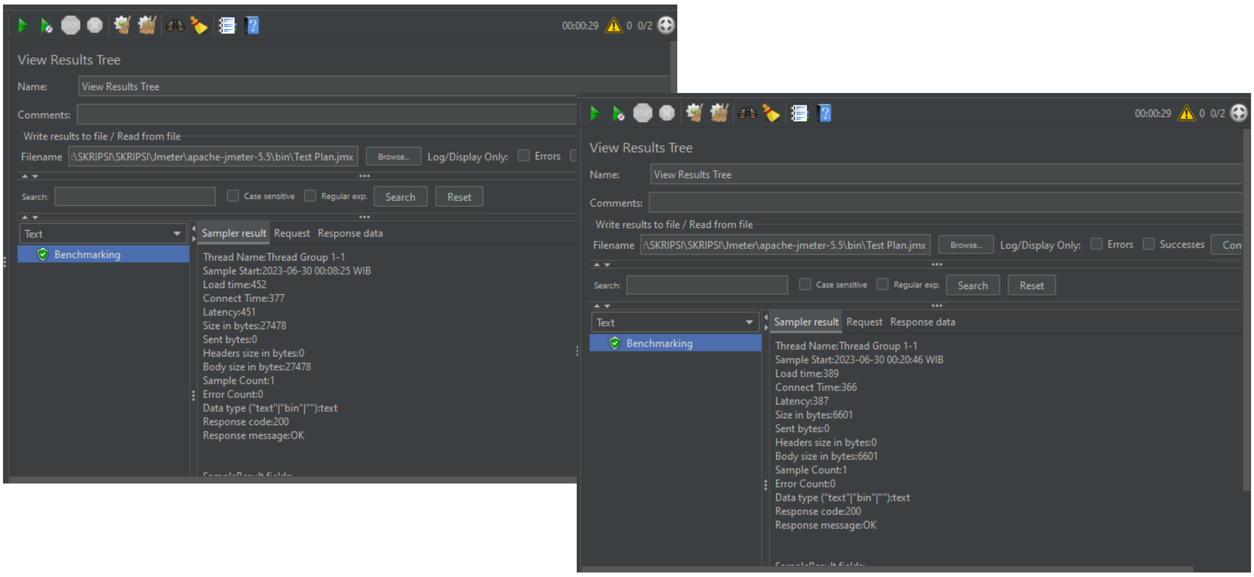
```
Thread Name:Thread Group 1-1
Sample Start:2023-06-29 22:47:04 WIB
Load time:427
Connect Time:377
Latency:426
Size in bytes:27478
Sent bytes:0
Headers size in bytes:0
Body size in bytes:27478
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:OK
```

00:00:29 0 0/2
View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename :\SKRIPSI\SKRIPSI\Jmeter\apache-jmeter-5.5\bin\Test Plan.jmx
Browse...
Log/Display Only: □ Errors □ Successes
Con
Search:
□ Case sensitive □ Regular exp.
Search
Reset
Text
Benchmarking
Benchmarking
Sampler result
Request
Response data

```
Thread Name:Thread Group 1-1
Sample Start:2023-06-30 00:14:08 WIB
Load time:386
Connect Time:363
Latency:384
Size in bytes:6601
Sent bytes:0
Headers size in bytes:0
Body size in bytes:6601
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:OK
```

Fig.9: Benchmarking for 'customers' and 'Projects' tables after encryption

Figure 9 shows the benchmarking results after implementing protection techniques for the 'customers' tables. The benchmarking results showed that the load time required to load the database with the same data transaction of 27478 bytes was 427 ms, while the connect time needed to make connections to the database was 377 ms, with a latency of 426 ms. Load time, connect time, and latency results from benchmarking results after encryption are higher than benchmark results before encrypting. The figure also shows the results of benchmarking performed after implementing protection techniques for the 'Projects' table. The benchmarking results showed that the load time required to load a database with the same data transaction of 6601 bytes was 386 ms, while the connect time needed to connect with the database was 363 ms, with a latency of 384 ms. The load time, connect time, and latency outcomes from the benchmarking results for the 'Projects' table after encryption, as in the Customers table, are higher than those from the Benchmark results before encrypting. This shows that MySQL TDE encryption affects the performance of the system. The average load time increased by 3.4%, from 409 ms to 427 ms for 'customers' tables and from 377 ms to 386 ms for project tables. The average connectivity time was increased by 6.3%, with an increase from 349 ms to 377 ms for the 'customers' table and a rise from 347 ms to 363 ms in the projects table. In addition, there was an increase in latency, where the average latency increased by 3.4%, with the latency increasing from 408 ms to 426 ms for 'customers' tables and an increase from 375 ms to 384 ms for project tables. The three aspects compared in the benchmarking process for table levels showed an insignificant decrease in performance. For the benchmarking test of encryption at the table space level, we will also benchmark by performing the "Select" query for the "customers" and "Projects" tables. Here are the benchmarking results for the 'customers' tables after encryption at the table space level:



Name: View Results Tree

Filename :\SKRIPSI\SKRIPSI\Jmeter\apache-jmeter-5.5\bin\Test Plan.jmx

```
Thread Name:Thread Group 1-1
Sample Start:2023-06-30 00:08:25 WIB
Load time:452
Connect Time:377
Latency:451
Size in bytes:27478
Sent bytes:0
Headers size in bytes:0
Body size in bytes:27478
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:OK
```

Name: View Results Tree

Filename :\SKRIPSI\SKRIPSI\Jmeter\apache-jmeter-5.5\bin\Test Plan.jmx

```
Thread Name:Thread Group 1-1
Sample Start:2023-06-30 00:20:46 WIB
Load time:389
Connect Time:366
Latency:387
Size in bytes:6601
Sent bytes:0
Headers size in bytes:0
Body size in bytes:6601
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:OK
```

Fig.10: Benchmarking for ‘customers’ and ‘Projects’ for table space level encryption

Figure 4.40 shows the benchmarking results performed after encryption at the table space level for the ‘customers’ table. The benchmarking results showed that the load time required to load a database with the same data transaction of 27478 bytes was 452 ms, while the connect time needed to connect with the database was 377 ms, with a latency of 451 ms. The figure also shows the benchmarking results performed after encryption at the table space level for the ‘Projects’ table. The benchmarking results showed that the load time required to load the database with the same data transaction of 6601 bytes was 389 ms, while the connect time needed to make connections to the database was 366 ms, with a latency of 387 ms. Load time, connect time, and latency results from benchmarking results after encryption at table space level for tables ‘customers and projects are higher than benchmark results before encrypting. The impact of the table space encryption process on the database can be seen from its average load time increase of 6.8%, with an increase from 409 ms to 452 ms for ‘customers’ tables and a rise from 377 ms to 389 ms for project tables, the average connectivity time increased by 6.7%, with an increase from 349 ms to 377 ms for 'customers' tables and a rise from 347 ms to 366 ms for project tables. In addition, there was an increase in latency, where the average latency increased by 6.9%, with the latency increasing from 408 ms to 451 ms for 'customers' tables and an increase from 375 ms to 387 ms for project tables. The three aspects compared in the benchmarking process for table and table space levels showed an insignificant decrease in performance. But the difference between doing encryption for a particular table and table-space encryption. The average increase in load time on table-level encryption was 3.4 percent to 6.8 percent, while average increases in connect time increased from 6.3 percent to 6.7 percent, and average latency increased from 3.4 percent to 6.9 percent. These results indicate that the ability of MySQL TDE to perform encryption at the table level for a particular table will be able to improve the performance efficiency of the database system used, as encrypting can be done specifically for certain tables that store sensitive data in the form of data-at-rest that suits the needs of the company.

4.5. Evaluation

The research is aimed at seeing whether the protection techniques implemented for the corporate database can meet the needs of the company and address the problems experienced by the company related to the security of the database and the extent to which the impact of the protection technologies implemented on the corporative database on the performance of the databases. The research results show that implementing the selected protection techniques, namely MySQL TDE, can protect the data in the database and maintain its high availability. Database encryption protection techniques using MySQL TDE show that the implementation of MySQL TDE has been proven to effectively encrypt data so that when an unauthorized party accesses the database, the data cannot be read or understood because the data has been encoded. However, the success of implementing protection techniques also affects the performance of the database system. The benchmarking results show increased load time, connect time, and latency. However, the increase was insignificant—below 10%—so it is still acceptable, especially compared to the security and high availability benefits obtained by databases. Overall, implementing protection techniques has proven to protect the database well and maintain high availability, even if there is a slight decrease in database performance.

The research aligns with previous studies that have successfully protected databases using other methods. In terms of security, previous research has demonstrated that using the AES algorithm as an encryption method for databases can provide a high level of security (Handoyo & Subakti, 2020). In this study, implementing MySQL TDE with the AES-256 encryption algorithm has also effectively protected MySQL databases with a high level of security. In terms of efficiency, previous research has shown that the AES algorithm for encryption exhibits good performance in terms of speed and efficiency (Ghosh, 2020). Implementing MySQL TDE in this study, utilizing the AES-256 algorithm, also demonstrates its ability to efficiently protect the database, as evidenced by a performance decrease of less than 10% during the benchmarking process.

5. Conclusion

Based on the conducted research, it can be concluded that MySQL database protection using MySQL Enterprise Transparent Data Encryption (TDE) at PT ABC demonstrates the following:

- Encrypting the database using MySQL TDE helps PT ABC protect the data within its database. Database protection using encryption with MySQL TDE effectively addresses PT ABC's concerns regarding data security. The database protection technique using encryption with MySQL TDE proves to safeguard the data within the database effectively. Therefore, when unauthorized individuals access the database, the accessed data remains unreadable and incomprehensible due to encryption.
- The implemented encryption using MySQL TDE efficiently protects the database.

Benchmarking shows that encryption using MySQL TDE impacts system performance, but the impact is insignificant. The benchmarking results indicate that implementing encryption using MySQL TDE increases the database system's load time, connect time and latency. However, the benchmarking results do not show a significant decrease in database system performance, making it acceptable and considered good efficiency. The observed increase in load time, connect time, and latency from the benchmarking results is insignificant, so it is acceptable, especially compared to the data security benefits obtained for the database.

As for the findings obtained from this research, future research on database protection could include the following aspects:

- Subsequent research can explore encrypting the MySQL database using encryption tools from other MySQL Enterprise Edition offerings, such as MySQL Enterprise Encryption, and compare it with MySQL Transparent Data Encryption. The research can focus on comparing the effectiveness and efficiency of both encryption techniques provided by MySQL Enterprise Edition.
- Further research can explore using protection techniques other than encryption, such as incorporating authentication mechanisms using MySQL Enterprise Authentication and comparing its effectiveness with encryption as a protection technique. The research can focus on comparing the effectiveness of encryption with other selected protection techniques.
- Future research can explore databases other than MySQL that provide similar tools as MySQL TDE, such as Microsoft SQL Server with its Microsoft SQL Server TDE feature, and evaluate the implementation conducted. This research can demonstrate how TDE can be effectively utilized across various database systems, not just MySQL.

Acknowledgements

This work is supported by Universitas Multimedia Nusantara, especially the Information System Department.

References

Adi, Puput Dani Prasetyo, and Akio Kitagawa. "Performance evaluation WPAN of RN-42 bluetooth based (802.15. 1) for sending the multi-sensor LM35 data temperature and raspBerry pi 3 Model B for the database and internet gateway." International Journal of Advanced Computer Science and Applications (IJACSA) 9, no. 12 (2018): 612-620.

Ali, M.R., Pal, D., Das, A. and Roychowdhury, D., 2022. HARPOCRATES: An Approach Towards Efficient Encryption of Data-at-rest. Cryptology ePrint Archive.

Al-Zahrani, A. and Al-Hebbi, M., 2022. Big Data Major Security Issues: Challenges and Defense Strategies. Tehnički glasnik, 16(2), pp.197-204..

Billah, M.M.T., 2020. Benchmarking dalam Islam (Ikhtiar dalam peningkatan mutu pendidikan). Jurnal Manajemen Pendidikan, 1(1), pp.1-15.

DB-Engines. (2023). DB-Engines Ranking. Retrieved from https://db-engines.com/en/ranking on October 9, 2023.

Elmasri, Ramez, Navathe, & Shamkant. (2017). Fundamental of Database System, 7th ed. Pearson Education.

Ghosh, A., 2020. Comparison of encryption algorithms: AES, Blowfish and Twofish for security of wireless networks. International Research Journal of Engineering Technology, 7, pp.4656-4658.

Gosal, R. and Rustam, A., 2022. Perancangan Sistem Informasi Inventory Berbasis Web Pada Gudang Di Pt. Spin Warriors. Aisyah Journal Of Informatics and Electrical Engineering (AJIEE), 4(1), pp.27-32.

Győrödi, C.A., Dumşe-Burescu, D.V., Zmaranda, D.R. and Győrödi, R.Ş., 2022. A Comparative Study of MongoDB and Document-Based MySQL for Big Data Application Data Management. Big Data and Cognitive Computing, 6(2), p.49.

Handoyo, J. and Subakti, Y.M., 2020. Keamanan Dokumen Menggunakan Algoritma Advanced Encryption Standard (AES). Jurnal SITECH: Sistem Informasi Dan Teknologi, 3(2), pp.143-152.

Helmud, E., 2021. Optimasi Basis Data Oracle Menggunakan Complex View Studi Kasus: PT. Berkat Optimis Sejahtera (PT. BOS) Pangkalpinang. INFORMANIKA, 7(01).

Kurniawan, T.B., 2020. Perancangan sistem aplikasi pemesanan makanan dan minuman pada cafetaria no caffe di Tanjung Balai Karimun menggunakan bahasa pemograman PHP Dan MySQL. Jurnal Tikar, 1(2), pp.192-206.

Manohar, N. and Kumar, P.V., 2020, May. Data encryption & decryption using steganography. In 2020 4th international conference on intelligent computing and control systems (ICICCS) (pp. 697-702). IEEE.

Mudrikah, F.Z.A. and Aditya, B., 2022, November. Design of a Geographic Information System for Forest and Land Fires Based on a Real-Time Database on Microservices Infrastructure. In 2022 IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS) (pp. 1-6). IEEE.

MySQL. (2023). InnoDB Cluster. Retrieved from MySQL.com.

MySQL. (2023). MySQL Enterprise Backup. Retrieved from MySQL.com: https://www.mysql.com/products/enterprise/backup.html

MySQL. (2023). MySQL Enterprise Transparent Data Encryption (TDE). Retrieved from MySQL.com: https://www.mysql.com/products/enterprise/tde.html

MySQL. (2023). MySQL Enterprise Transparent Data Encryption (TDE). Retrieved from MySQL.com: https://www.mysql.com/products/enterprise/tde.html

Rafique, A., Van Landuyt, D., Beni, E.H., Lagaisse, B. and Joosen, W., 2021. CryptDICE: Distributed data protection system for secure cloud data storage and computation. Information Systems, 96, p.101671.

Shiyal, B., 2021. Introduction to azure synapse analytics. In Beginning Azure Synapse Analytics: Transition from Data Warehouse to Data Lakehouse (pp. 49-68). Berkeley, CA: Apress.

Sibyan, H., 2017. Implementasi Enkripsi Basis Data Dengan Algoritma Md5 (Message Digest Algorithm 5) dan Vigenere Cipher. Jurnal PPKM, 1, pp.114-121.

Steele, T., Patten, C. and Kottmann, D., 2020. Black Hat Go: Go Programming For Hackers and Pentesters. No Starch Press.

Stragapede, G., Vera-Rodriguez, R., Tolosana, R. and Morales, A., 2023. BehavePassDB: public database for mobile behavioral biometrics and benchmark evaluation. Pattern Recognition, 134, p.109089.

Suhaemin, A. and Muslih, M., 2023. Karakteristik Cybercrime di Indonesia. Edulaw: Journal of Islamic Law and Jurisprudance, 5(2), pp.15-26.

Susilo, G., 2017. Keamanan basis data pada sistem informasi di era global. Transformasi, 12(2).

Vicarte, J.R.S., Flanders, M., Paccagnella, R., Garrett-Grossman, G., Morrison, A., Fletcher, C.W. and Kohlbrenner, D., 2022, May. Augury: Using data memory-dependent prefetchers to leak data at rest. In 2022 IEEE Symposium on Security and Privacy (SP) (pp. 1491-1505). IEEE.

Wiharto, Y. and Irawan, A., 2018. Enkripsi data menggunakan advanced encryption standard 256. Jurnal Kilat, 7(2), pp.91-99.

Yani, A., Saputra, B. and Jurnal, R.T., 2018. Rancang Bangun Sistem Informasi Evaluasi Siswa Dan Kehadiran Guru Berbasis Web. Petir, 11 (2).